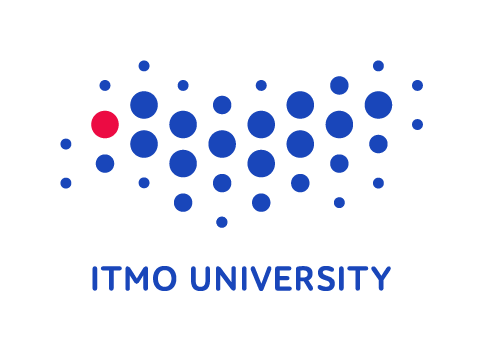
Санкт-Петербургский Национальный Исследовательский

Университет Информационных Технологий, Механики и Оптики

Факультет Программной Инженерии и Компьютерной Техники



Варианты №: 1019

Лабораторная работа № 5

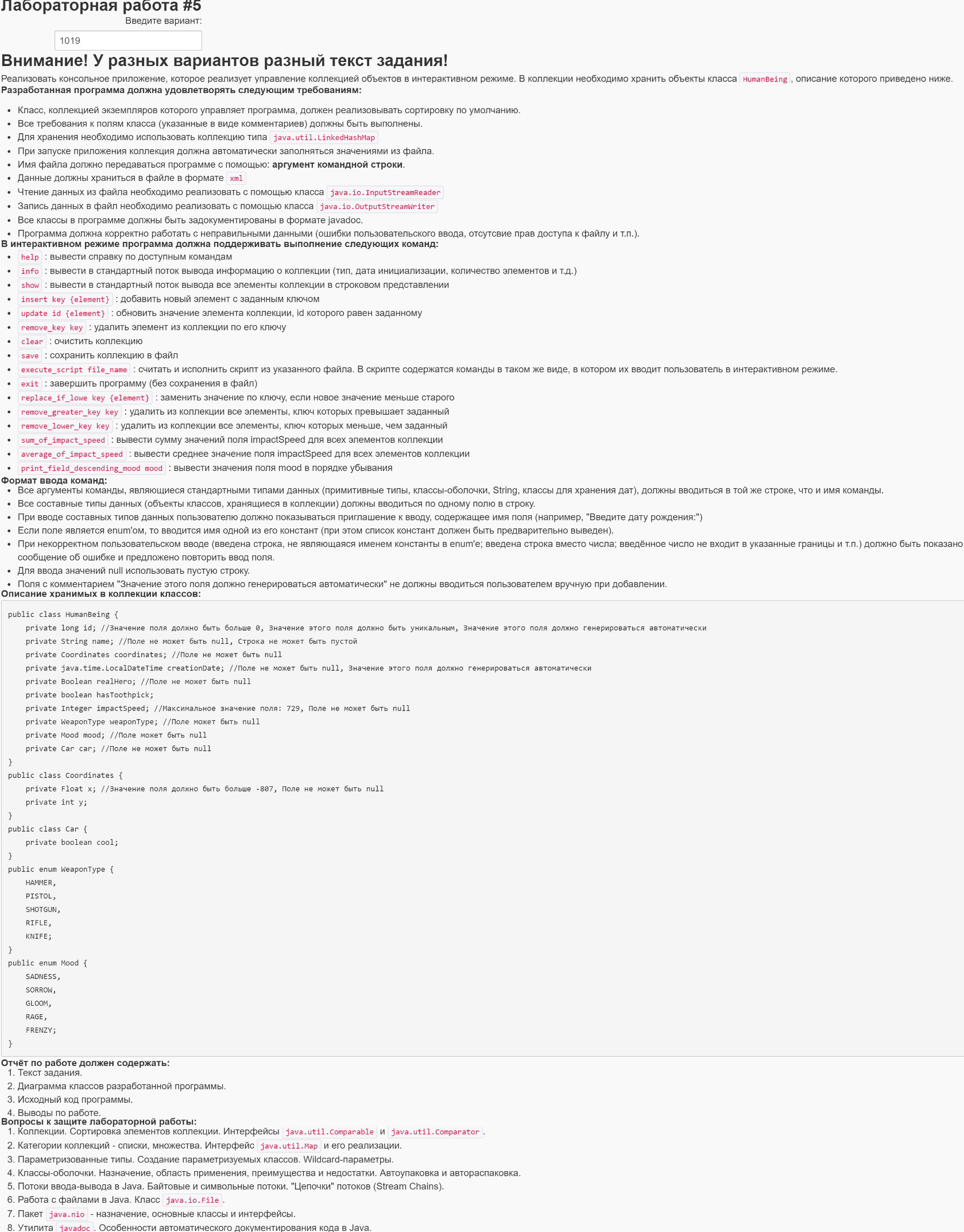
по дисциплине

*«Программирование»*

Выполнил: Провоторов Александр Владимирович  
Студент группы: P3110

Преподаватель: Николаев Владимир Вячеславович

# ****Текст Задания****



# ****Диаграмма классов реализованной объектной модели.****

**См на отдельном листе (Приложение 1)**

# ****Исходный код программы.****

## ****Class Main****

**import** Exceptions.RightException;  
**import** Command.CommandShell;  
**import** MyOwnClasses.HumanBeing;  
**import** MyOwnClasses.HumanList;  
**import** Tools.Parser;  
  
**import** javax.xml.bind.JAXBException;  
**import** java.io.BufferedReader;  
**import** java.io.File;  
**import** java.io.IOException;  
**import** java.io.InputStreamReader;  
**import** java.util.\*;  
  
**import static** Tools.Converter.*convertToList*;  
**import static** Tools.Converter.*сonvertToLinkedHashMap*;  
  
**public class** Main {  
 **public static void** main(String []args) **throws** IOException {  
 CommandShell commandShell = **new** CommandShell();  
 InputStreamReader reader = **new** InputStreamReader(System.***in***);  
 BufferedReader bufferedReader = **new** BufferedReader(reader);  
 **boolean** b = **true**;  
 String path = **""**;  
  
 **try** {  
 Parser Parser = **new** Parser();  
 path = args[0];  
 **if** (!(path == **""**)) Parser.parse(path);  
 System.***out***.println(**"Файл был найден и конвертирован в коллекцию"**);  
 } **catch** (ArrayIndexOutOfBoundsException e) {  
 File file = **new** File(**"New\_File"**);  
 path = file.getName();  
 System.***out***.println(**"Название файла было введено неправильно или такого файла не существует, файл \""** + path +**"\" был создан"**);  
 } **catch** (RightException e) {  
 File file = **new** File(**"New\_File"**);  
 path = **"New\_File"**;  
 System.***out***.println(**"Возникла ошибка с правами доступа к файлу, файл \""** + path + **"\" был создан"**);  
 } **catch** (JAXBException e) {  
 System.***out***.println(**"XML файл был сгенерирован неверно"**);  
 }  
 LinkedHashMap<Integer, HumanBeing> human = **new** LinkedHashMap<Integer, HumanBeing>();  
  
 **try**{  
 human = *сonvertToLinkedHashMap*((Parser.*getHumanList*()));  
 } **catch** (NullPointerException e){}  
 HumanList humanList = **new** HumanList();  
 humanList.creationDate\_Now();  
 humanList.setHumanBeings(*convertToList*(human));  
  
 **while** (b || reader.ready()) {  
 System.***out***.print(**"Введите команду: "**);  
 String command = bufferedReader.readLine();  
 human = commandShell.Analyze(humanList, human, command, path, b);  
 humanList.setHumanBeings(*convertToList*(human));  
 }  
 }  
}

## Class Parser

**package** Tools;  
  
**import** Exceptions.ExistanceException;  
**import** Exceptions.HumanValueException;  
**import** Exceptions.RightException;  
**import** MyOwnClasses.HumanList;  
  
**import** javax.xml.bind.JAXBContext;  
**import** javax.xml.bind.JAXBException;  
**import** javax.xml.bind.Unmarshaller;  
**import** java.io.File;  
  
**import static** Tools.Checker.*checkHumanList*;  
  
  
**public class** Parser{  
 **private static** HumanList *humanList*;  
 **public void** parse(String string) **throws** RightException, JAXBException, ExistanceException {  
  
 File file = **new** File(string);  
 **if** (!file.canRead() && !file.canWrite() && file.exists()) **throw new** RightException(**"Возникла ошибка с правами файла"**);  
 **if** (!file.exists()) **throw new** ExistanceException(**"Данного файла не существует"**);  
 JAXBContext jaxbContext = JAXBContext.*newInstance*(HumanList.**class**);  
  
 Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller();  
  
 HumanList humanList = (HumanList) jaxbUnmarshaller.unmarshal(file);  
  
 **this**.*humanList* = humanList;  
  
 **try**{  
 **for** (**int** i = 0; i < humanList.getHumanBeings().size(); i++) humanList.getHumanBeing(i).setId(i+1);  
 }**catch** (NullPointerException e){}  
  
 **try** {  
 *checkHumanList*(humanList);  
 } **catch** (HumanValueException e){  
 e.printStackTrace();  
 }  
 }  
 **public static** HumanList getHumanList(){  
 **return** *humanList*;  
 }  
}

## Class ComandShell

**package** Command;  
  
**import** Exceptions.HumanValueException;  
**import** MyOwnClasses.HumanBeing;  
**import** MyOwnClasses.HumanList;  
  
**import** java.util.\*;  
  
**public class** CommandShell {  
 **public** CommandShell() {}  
 **public** LinkedHashMap<Integer, HumanBeing> Analyze(HumanList humanList, LinkedHashMap<Integer, HumanBeing> human, String command, String path, **boolean** b) {  
 HashMap<String, Command> commandList = **new** LinkedHashMap<>();  
 StringTokenizer tokenizer = **new** StringTokenizer(command);  
 commandList.put(**"save"**, **new** Save(human, command, humanList));  
 commandList.put(**"info"**, **new** Info(human, command, humanList));  
 commandList.put(**"exit"**, **new** Exit(human, command, humanList));  
 commandList.put(**"help"**, **new** Help(human, command, humanList));  
 commandList.put(**"show"**, **new** Show(human, command, humanList));  
 commandList.put(**"clean"**, **new** Clean(human, command, humanList));  
 commandList.put(**"update"**, **new** Update(human, command, humanList));  
 commandList.put(**"insert"**, **new** Insert(human, command, humanList));  
 commandList.put(**"remove\_key"**, **new** RemoveKey(human, command, humanList));  
 commandList.put(**"execute\_script"**, **new** ExecuteScript(human, command, humanList));  
 commandList.put(**"replace\_if\_lowe"**, **new** ReplaceIfLowe(human, command, humanList));  
 commandList.put(**"remove\_lower\_key"**, **new** RemoveLowerKey(human, command, humanList));  
 commandList.put(**"remove\_greater\_key"**, **new** RemoveGreaterKey(human, command, humanList));  
 commandList.put(**"sum\_of\_impact\_speed"**, **new** SumOfImpactSpeed(human, command, humanList));  
 commandList.put(**"average\_of\_impact\_speed"**, **new** AverageOfImpactSpeed(human, command, humanList));  
 commandList.put(**"print\_field\_descending\_mood"**, **new** PrintFieldDescendingMood(human, command, humanList));  
 **try** {  
 String token = tokenizer.nextToken();  
 human = commandList.get(token.toLowerCase()).execute(human, command, humanList, path, b);  
 } **catch** (StringIndexOutOfBoundsException e) { System.***out***.println(**"Команда была введена неверно"**); }  
 **catch** (ArithmeticException e) { System.***out***.println(**"Лист HumanList пуст"**); }  
 **catch** (NullPointerException e) { System.***out***.println(**"Команда была введена неверно"**); }  
 **catch** (HumanValueException e) { System.***out***.println(**"Введите правильный тип данных"**); }  
 **return** human;  
 }  
}

# ****Результат работы программы.****

|  |
| --- |
| java18 -jar Lab5.jar Exercise.xml |
| Возникла ошибка с правами доступа к файлу, файл"New\_File" был создан |
| Введите команду: exit |
| Конец программы |
| s284695@helios:/home/s284695/Lab5$ java18 -jar Lab5.jar Exercise.xml |
| Возникла ошибка с правами доступа к файлу, файл "New\_File" был создан |
| Введите команду: save |
| Коллекция сохранена в файл New\_File |
| Введите команду: exit |
| Конец программы |
| s284695@helios:/home/s284695/Lab5$ cat New\_File |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?> |
| <humanList/> |
| s284695@helios:/home/s284695/Lab5$ chmod 700 Exercise.xml |
| s284695@helios:/home/s284695/Lab5$ java18 -jar Lab5.jar Exercise.xml |
| Файл был найден и конвертирован в коллекцию |
| Введите команду: show |
| Коллекция: [HumanBeing{id=1, name='a', coordinates=Coordinates{x=82.0, y=1}, creationDate=2020-03-06T01:02:31.566, realHero=true, hasToothpick=true, impactSpeed=20, weaponType=SHOTGUN, mood=RAGE, car=Car{cool=true}}, HumanBeing{id=2, name='a', coordinates=Coordinates{x=82.0, y=2}, creationDate=2020-03-06T01:02:31.572, realHero=true, hasToothpick=true, impactSpeed=20, weaponType=SHOTGUN, mood=FRENZY, car=Car{cool=true}}, HumanBeing{id=3, name='2', coordinates=Coordinates{x=2.0, y=2}, creationDate=2020-03-06T01:02:31.573, realHero=true, hasToothpick=true, impactSpeed=22, weaponType=HAMMER, mood=SADNESS, car=Car{cool=true}}] |
| Введите команду: info |
| Type: LinkedHashMap |
| Initialization date: 2020-03-06 01:02 |
| Number of elements: 3 |
| Введите команду: print\_descending\_mood |
| Команда была введена неверно |
| Введите команду: help  help : вывести справку по доступным командам  info : вывести в стандартный поток вывода информацию о коллекции (тип, дата инициализации, количество элементов и т.д.)  show : вывести в стандартный поток вывода все элементы коллекции в строковом представлении  insert key {element} : добавить новый элемент с заданным ключом  update id {element} : обновить значение элемента коллекции, id которого равен заданному  remove\_key key : удалить элемент из коллекции по его ключу  clear : очистить коллекцию  save : сохранить коллекцию в файл  execute\_script file\_name : считать и исполнить скрипт из указанного файла. В скрипте содержатся команды в таком же виде, в котором их вводит пользователь в интерактивном режиме.  exit : завершить программу (без сохранения в файл)  replace\_if\_lowe key {element} : заменить значение по ключу, если новое значение меньше старого  remove\_greater\_key key : удалить из коллекции все элементы, ключ которых превышает заданный  remove\_lower\_key key : удалить из коллекции все элементы, ключ которых меньше, чем заданный  sum\_of\_impact\_speed : вывести сумму значений поля impactSpeed для всех элементов коллекции  average\_of\_impact\_speed : вывести среднее значение поля impactSpeed для всех элементов коллекции  print\_field\_descending\_mood mood : вывести значения поля mood в порядке убывания  Введите команду: average\_of\_impact\_speed  20  Введите команду: save  Коллекция сохранена в файл Exercise.xml  Введите команду: exit  Конец программы |

# ****Вывод****

**В ходе выполнения лабораторной работы я познакомился с некоторыми видами коллекций, категориями коллекций. Также затронул потоки ввода-вывода, поработал с файлами в Java и попробовал задокументировать файл в Javadoc.**